Spring 5-25-2021

Fake News Analysis and Graph Classification on a COVID-19 Twitter Dataset

Kriti Gupta

Fake News Analysis and Graph Classification on a COVID-19 Twitter Dataset

A Project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Kriti Gupta

May 2021

The Designated Project Committee Approves the Project Titled

Fake News Analysis and Graph Classification on a COVID-19 Twitter Dataset

by

Kriti Gupta

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSÉ STATE UNIVERSITY

May 2021

Dr. Katerina Potika	Department of Computer Science
Dr. Fabio DiTroia	Department of Computer Science
Dr. Magdalini Eirinaki	Department of Computer Engineering

ABSTRACT

Fake News Analysis and Graph Classification on a COVID-19 Twitter Dataset

by Kriti Gupta

Earlier researches have showed that the spread of fake news through social media can have a huge impact to society and also to individuals in an extremely negative way. In this work we aim to study the spread of fake news compared to real news in a social network. We do that by performing classical social network analysis to discover various characteristics, and formulate the problem as a binary classification, where we have graphs modeling the spread of fake and real news. For our experiments we rely on how news are propagated through a popular social media services such as Twitter during the pandemic caused by the COVID-19 virus. In the past, several other approaches classify news as fake or real by deploying various graph embedding techniques and deep learning techniques.

In this project we focus on developing a dataset that contains tweets specific to COVID-19 by performing initially text mining on the content of the tweet. Further, we create graphs of the fake and real news along with their retweets and followers and work on the graphs. We perform social network analysis and compare their characteristics. We study the propagation of fake and real news among users using community detection algorithms on the graphs. Finally, we create a model by deploying the Weisfeiler Lehman graph kernel for graph classification on our labeled dataset. The model is able to predict whether a new article is real or fake based on how the corresponding graph of the retweets and followers are connected.

Keywords - Graph kernels, community detection, COVID-19, Weisfeiler Lehman kernel, graph classification, fake news

ACKNOWLEDGMENTS

It is my privilege and utmost duty to acknowledge the learned Professors who have spared their valuable time guiding me in this project and boosting my morale, time and again, while working on it.

I would like to express my deep and sincere gratitude to my project advisor Professor Katerina Potika for giving me the opportunity to work on the project, 'Fake News Analysis and Graph Classification on a COVID-19 Twitter Dataset', which would be very beneficial to the masses during these trying times of Covid-19. It would help them in avoiding their getting misguided by the day-to-day fake news making the rounds unabatedly. I am highly grateful to Professor Potika for being by my side whenever I needed her help. Her dynamism, vision, sincerity, motivation and above all, affectionate methodology have greatly inspired me.

I would also like to thank Professor Magdalini Eirinaki, who very diligently showed me the path for the collection of various relevant data much needed in the functioning of the project. She was a big support to me.

Also, I am deeply thankful to Professor Fabio DiTroia who supported me all the way through my project by his vastly endowed knowledge and guidance.

I am very fortunate to have these professors as my mentors with whose help and support I could complete my project.

TABLE OF CONTENTS

CHAPTER

1 Introduction 1

1.1 Problem Definition 1

1.2 Motivations of this Research 1

2 Related Work 6

2.1 Text Mining 6

2.1.1 Latent Dirichlet Allocation (LDA) 6

2.2 Social Network Analysis 7

2.2.1 Community Detection 8

2.3 Machine Learning 8

2.3.1 Graph Classification 9

3 Methodology 11

3.1 Innovative and challenging aspects 11

3.2 Phases in the implementation plan 12

3.3 Dataset 13

3.3.1 FakeNewsNet 14

3.3.2 Data Pre-processing 16

3.4 Creation of the Graphs 17

3.4.1 Time Complexity Analysis 20

3.5 Weisfeiler-Lehman Kernel Approach 21

3.5.1 Computation 21

3.5.2 Time Complexity Analysis 22

4 Experimental Evaluation 24

4.1 Graph classification results 29

4.1.1 Classifiers 29

4.1.2 Steps for performing WL subtree kernel approach 33

5 Conclusions and Future Work 36

LIST OF REFERENCES 37

LIST OF TABLES

1 Overview of Fake News Graphs 14

2 Overview of Real News Graphs 15

3 Graph Classification Accuracy 35

LIST OF FIGURES

1	Examples of Fake News During the Covid-19 Pandemic.	2
2	Twitter Flagging Covid-19 Tweets.	3
3	Deaths Caused by Disinfectant Poisoning [1]	3
4	Example of LDA Topic Modelling [2]	7
5	Graph Classification Problem: Which one is real? Graph A or Graph B	11
6	Implementation Plan.	12
7	FakeNewsNet Architecture.	16
8	Tweet Description	17
9	Word Cloud	18
10	Tweet Graph.	19
11	Steps Involved in the Computation of Weisfeiler-Lehman Kernel [3]	22
12	Fake Community Detection Analysis	26
13	Real Community Detection Analysis	26
14	Degree Distribution for Fake News	27
15	Degree Distribution for Real News	27
16	Page Ranking for Fake News	28
17	Page Ranking for Real News	28
18	Graph Visualization for Fake News Propagation	29
19	Graph Visualization for Real News Propagation	30
20	SVM Trained From Samples of Two Classes [4]	31

21 Example of KNN Classification [5] 31

22 Random Forest Classification [6] 32

CHAPTER 1

Introduction

1.1 Problem Definition

A piece of information is called fake news, if it is fabricated without the use of verifiable facts and resources, and is presented as (true) news. Fake news is difficult to detect and it can cause a lot of harm, like anxiety, mental stress, change in thoughts and opinions of someone regarding a topic. Most of the time, the creation of fake news is deliberately done to disorient people or groups of people. However, most of the dissemination of fake news is done unconsciously. Figure 1 give examples of how fake news is being shared on Twitter and then even re-tweeted and commented, which further makes it look like real news.

A study performed in 2020 on 1000 participants on fake news [7], shows that people tend to share news easily without discerning whether it is true or false. However, if the same participants know which news is fake it is more likely that they share (true) news instead of fake ones. This indicates that it is highly desirable to detect fake news in order to ensure the spread of credible information only. BigTech industries such as Twitter and Facebook have integrated fake news detecting artificial intelligent (AI) technology on their social media platforms, so that, whenever someone shares some news that seems to be fake the AI flags it as 'unverifiable' [8]. By flagging the false content, it is observed that people disregard the news they are seeing and do not get intoxicated by the fake news spreaders. Figure 2 shows Twitter flagging a user's content because they mentioned a coronavirus related tweet.

1.2 Motivations of this Research

My motivation for this research work was derived by the damage caused by fake news in my surroundings. In the news I saw that accidental poisonings among the people increased after President Trump commented that drinking or injecting

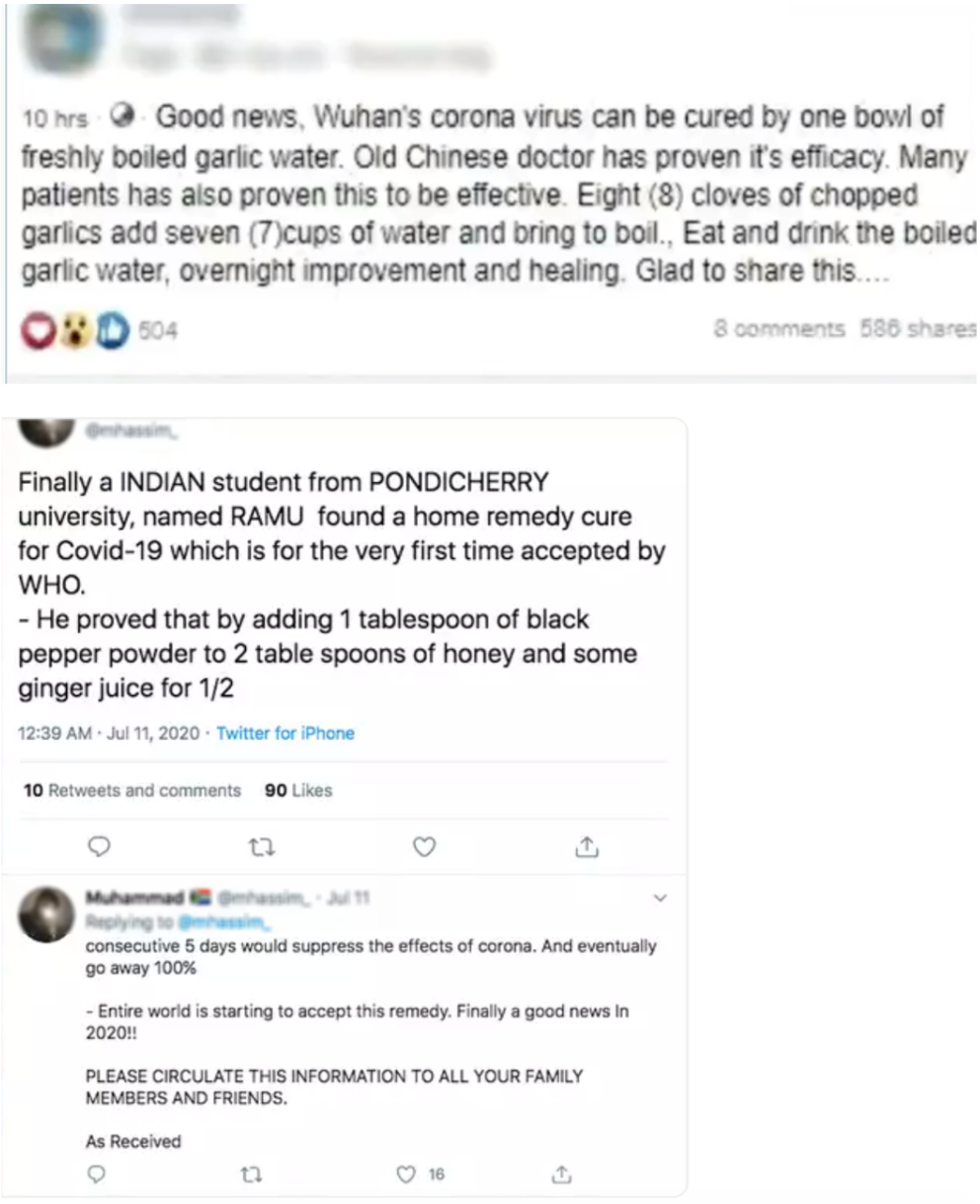

Figure 1: Examples of Fake News During the Covid-19 Pandemic.

disinfectants could cure from Covid 19 [1]. We can see from figure 3 how the cases doubled in the month of March and April in the year 2020 as compared to the year 2019. Further, this research is performed to ensure that:

- People should know the real facts and figures before sharing the information.

Figure 2: Twitter Flagging Covid-19 Tweets.

- It helps to improve one's credibility, by sharing true information.
- Misleading information can lead to hurting someone.
- A misinformed person or public can make wrong life altering decisions.

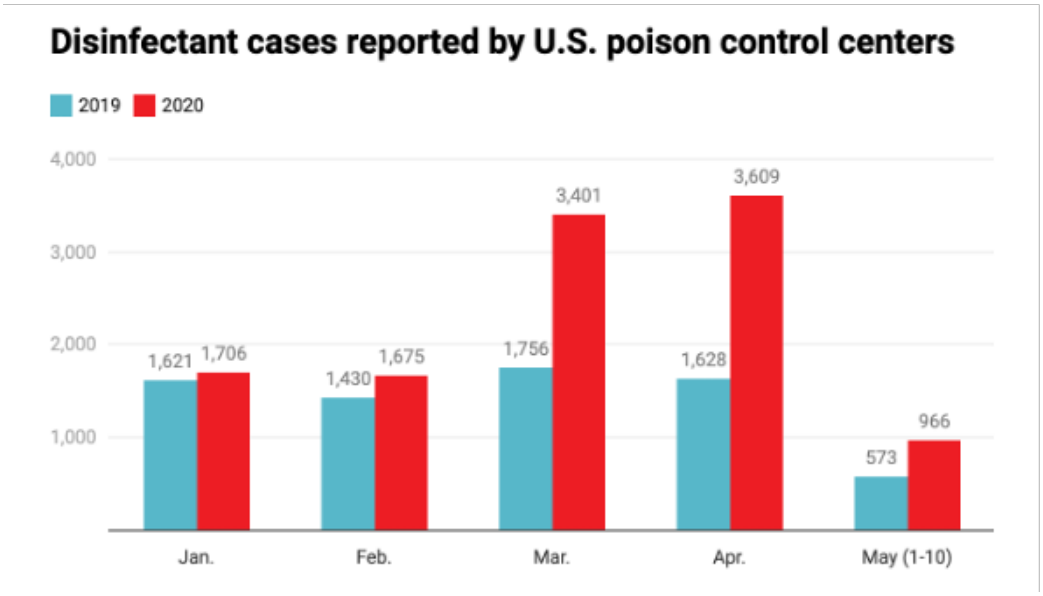

Figure 3: Deaths Caused by Disinfectant Poisoning [1]

The following research proves how devastating the impact of the spread of fake news has been on society during the COVID-19 outbreak. Ahmed et al. [9] analyzed the spread of the 5G conspiracy theory among the influenced users of Twitter during COVID-19 utilizing the Clauset-Newman-Moore algorithm in the United Kingdom (UK). In the 5G conspiracy theory, the social media users pointed out that 5G is the cause of the COVID-19 disease and that it is accelerating it's spread. This rumor soon turned into a concerning issue as the number of shared posts of images and videos increased on various social media platforms. Because of the spread of this fake news, people in UK started torching 5G towers. The communication towers of Nightingale hospital in Birmingham, UK was also set to fire during the spread of this conspiracy theory and it became difficult for the hospital staff to treat the patient's fighting the COVID-19 disease. The torching of the towers caused a lot of public health damage to the citizens of UK.

Previous researchers have analyzed the fake news on social media platforms, such as Twitter, by cross-verifying them with fact checking websites [10]. However, it is a tiresome task to cross-verify each and every tweet and may have erroneous results at times. Recently, some studies incorporated the spread of fake news using network analysis in order to detect fake news. As an example the study [11] investigates the various patterns of information propagation on Twitter, by exploring machine learning techniques, like the Weisfeiler-Lehman graph kernel for graph similarity, for determining the rumor veracity. Our proposed research is based on their observations [11] on the COVID-19 fake news.

In this research, we aim to develop several true/false information graphs from a twitter data set on fake information during the COVID-19 period. First, we will model our problem with information graphs. As the Weisfeiler-Lehman graph kernel method suggests we would be able to discern between false and true news by computing the

similarity between two graphs as the product of cosine-vectors of both their graph embeddings. Finally, in our research we would compare our proposed methodology with the existing research done for fake new analysis during COVID-19 [12].

More specifically for the model, each true/false information graph would be an Ego graph with nodes representing Twitter user Ids and edges representing the re-tweeted data. This modeling is an inspiration from [13]. There, they developed "cascades" of true/false information found on Twitter. Each cascade is defined as "a rumor-spreading pattern" which displays a continuous retweet chain having the same, single origin. On a similar note, researchers in [11] also deployed these cascade structures in their research work to determine rumor veracity.

For the classification, we will train a Weisfeiler-Lehman graph kernel algorithm on the constructed true/false information graphs. This will help us in observing the magnitude of similarity and differences between any two given graphs. Further, we would incorporate this supervised learning process into the contents of the tweets.

For testing our approach, we would generate multiple graphs for random fake news tweets and compare it's similarity with true and false news graphs. With this we will form a prediction accuracy for a news being retweeted on Twitter. Our research will help people in sharing only credible news on social media platforms. Thus, people will be able to make the right decisions for themselves and for their families and cause less stress and anxiety within a community during events such as the COVID-19 pandemic.

CHAPTER 2

Related Work

Since the dawn of December 2019, the novel coronavirus disease, also known as, COVID-19 started spreading rapidly and by the end of the year 2020, it affected nearly 74 million people world-wide [14]. In their research work, Garcia-Gasulla et al. [15] have quantitatively analyzed the impacts of COVID-19 on society's mobility, health, social and economic behavior. This data was used by them to inform the public and private sectors to make the effective and appropriate decisions. Their research includes sentiment distribution among the people on Twitter using the BERT [16] deep learning model and the STANZA tool [17] based on the factors of fear, anger, sadness, anticipation, joy, trust, disgust and surprise during COVID-19 outbreak. It has been found that as the news of the COVID-19 pandemic spread to people there was an increase in their fear, anger, sadness and anticipation.

2.1 Text Mining

Text mining is also known as data mining [18, 19]. It is referred to as the process of extracting useful data from a content. In the past, text mining has a variety of applications in research, government and business needs [20, 21]. Security applications utilize text mining for monitoring and analyzing the online content present on blogs, news. Text mining plays a tremendous role in sentiment analysis by detecting the emotions of people hidden in plain text written by them.

2.1.1 Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation (LDA) is a text mining generative probabilistic model described for the collections of the discrete data, presented by Blei et al. [2]. The motivation behind the LDA model is that to represent a document as arbitrary blend of latent topics. And each of these topics are denoted by a distribution on the words present in the topic.

They also presented an illustration of LDA topic modelling on real data consisting of 16000 documents from a section of TREC AP corpus. With the help of LDA topic modelling they were able to find four significantly large topics – 'Arts', 'Budgets', 'Children' and 'Education' in the corpus. These bags of words have an approximate distribution that tend to achieve a peak over 'k' possible topic values. Figure 4 is an example from this corpus. The different color of the words presents the four different categories that have been classified into by the LDA topic modelling algorithm.

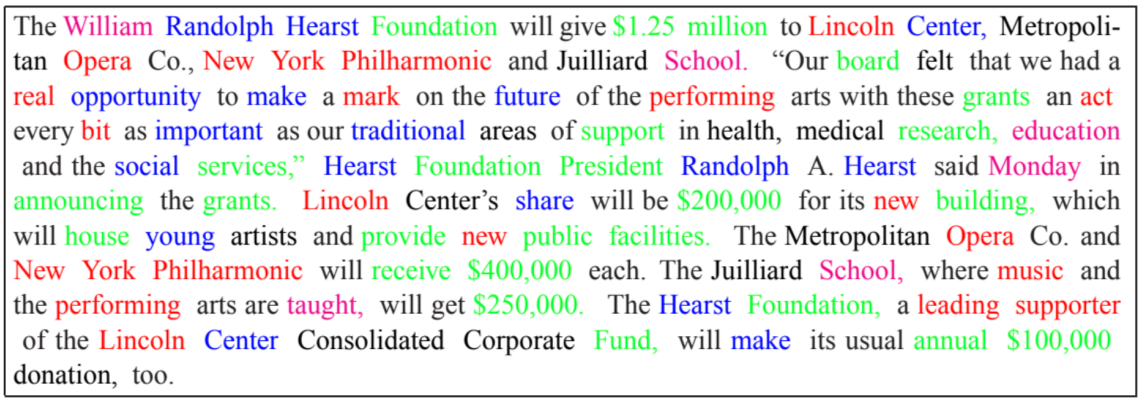

Figure 4: Example of LDA Topic Modelling [2]

Not only this, many researchers have utilized the LDA topic modeling approach in their research. One such work is that of Wu et al. [22], They have performed text mining on topic evolution by developing an LDA-based model. Their model is based on clarity algorithm that finds the hidden topic in a given text and then identifies the topic mutation over time. The clarity algorithm identifies the differences and similarities between topics, further which is used to determine the intensity trends of topics over a period of time.

2.2 Social Network Analysis

The process with which social structures are analyzed with the use of network and graph theory is called social network analysis [23]. It is characterized as a network

structure consisting of nodes and edges. Nodes can represent individual person, or an object within a network, while edges represent the interactions and relationships within these nodes. Social Network Analysis has been used to solve problems such as fake news analysis, sentiment analysis and similar problems. Community detection is a very important property in SNA and is explained the next section.

2.2.1 Community Detection

In social network analysis, community detection can be used to perform machine learning algorithms on given collection of graphs to detect groups which have identical properties. By analyzing the communities one can perceive the various reasons for which any two communities look alike. Research in [24] is a survey on community detection algorithms. They collected real world networks in a large scale from the benchmark dataset of Zachary's karate club. On this dataset they applied community detection algorithms such as leading eigen vector, edge betweenness, fast greedy, label propagation, multi-level, optimal modularity, spinglass, walktrap and infomap. Each algorithm worked differently in term of scalability, directed networks and overlapping communities.

2.3 Machine Learning

Machine learning problems usually revolve around a feature space which has objects represented as vectors, and the task is to train the machine to be able to distinguish the vectors between the vectors that belong to a positive subset and the vectors that belong to a negative subset [25]. It is quite difficult to select a local structure that would contribute to the feature space. Selecting the wrong local feature in a machine learning problem could result in combinatorial explosion, thus leading to NP-hard problem. In the following section, we aim to discuss the machine learning problem of graph classification using kernels [26, 27].

2.3.1 Graph Classification

The problem of graph classification has various applications in many domains. In order to find solution for this problem, the graph statistics or graph features needs to be calculated that would further help in discriminating between the graphs of different classes. One of the most important approach towards graph classification is the kernel approach like the support vector machines (SVMs).

Kernel methods [28, 29] involve the mapping of the graphs in the feature space and during the machine learning, the inner products of the vectors are considered. A 'kernel' is defined as the function which gives this inner product. The kernel method proves to be very efficient even in high dimensional spaces. In general, to define a kernel between any two random graphs, random walk on the vertex of the product graph consisting of those two graphs is used [30].

One such graph kernel is the Weisfeiler-Lehman (WL) Kernel. The WL kernel tests the isomorphism between two given graphs. One of the most important and unique property of the WL kernel is that is includes the node attribute, which can be referred to as node tags. A cascade of these node tags help in iterating the graph without involving any extra information such as the node identities. Rosenfeld et al. [11] performed the WL kernel approach on a dataset of 126,000 cascades containing 4.5 million tweets prepared by Vosoughi et al. [13]. They have presented in their research work is that of misinformation detection. With the help of graph kernels they extracted the topological information from the Twitter cascades. Then training is performed on the predictive models which do not have information on user identities, language and time, thus proving that that the diffusion patterns are highly informative on the truthfulness of an information. Their research proves that with the right aggregation the collective sharing pattern of the population could reveal that the information that exists among them is true or false.

A similar observation has been made by Neumann et al. [31]. In their research, authors introduce the propagation-kernels which is a graph-kernal framework. The propagation kernels are used to monitor how information spreads in a given set of graphs.

CHAPTER 3

Methodology

This research takes into consideration the creation of collection of graphs from Twitter tweets. Given this dataset of collection of graphs G, where $G = G_1, G_2, G_3, ..., G_{\mathrm{N}}$, and each G_{i} defined as, $G_{\mathrm{i}} = (V_{\mathrm{i}}, E_{\mathrm{i}})$, has vertices V_{i} and edges E_{i}. This graph classification problem focuses on categorizing unlabeled graphs into two categories: fake news and real news. Figure 5 shows two graphs which are identical to each other, the problem is to identify the differences between them to be able to categorize them into two different categories. First, we will provide an input dataset to train our model. Then, when we provide an unlabeled graph as input, our model should be able to predict the correct category to which the graph should belong to.

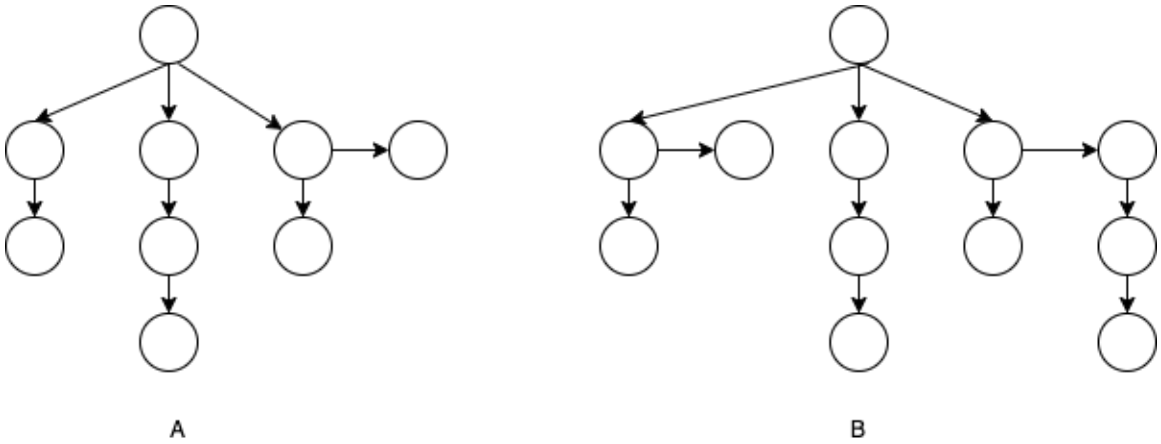

Figure 5: Graph Classification Problem: Which one is real? Graph A or Graph B

3.1 Innovative and challenging aspects

1. Re-hydrating the COVID-19 Twitter IDs and dividing them into two classes for fake and true news analysis

2. Developing an algorithm to generate graph embeddings and modeling them as Twitter IDs being the nodes and the content of the Tweet as the edge

3. Following a supervised learning process to incorporate in the Weisfeiler-Lehman graph kernels approach the contents of the tweets.

3.2 Phases in the implementation plan

1. Pre-processing of the data
2. Divide the pre-processed data into two classes: true/fake
3. Create graph from the given information
4. Methodology involving techniques for supervised-learning
5. Comparison with the existing approaches

The implementation plan is given by figure 6:

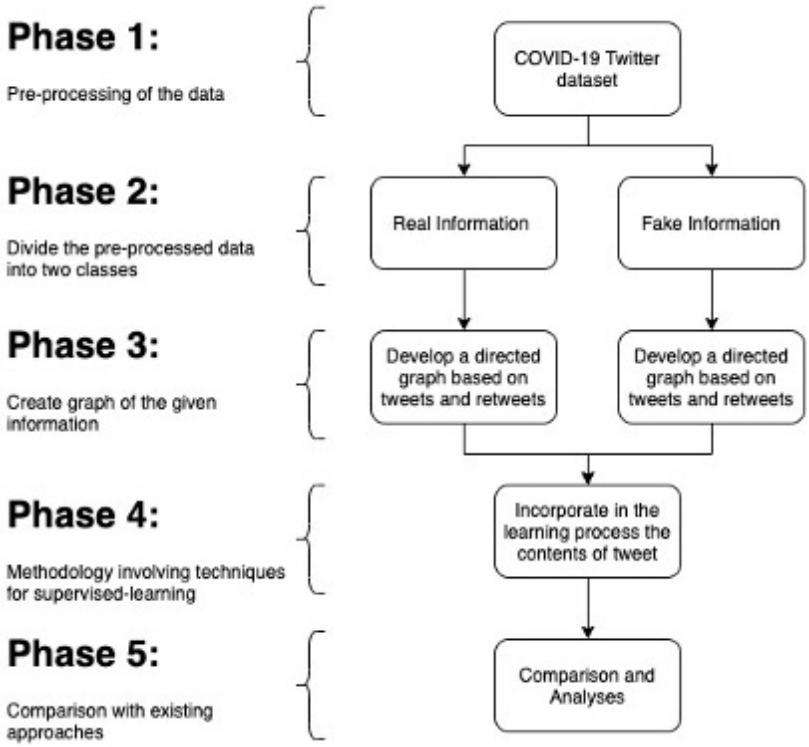

Figure 6: Implementation Plan.

The explanation of each phase in the implementation plan is as follows:

- Phase 1: Pre-processing of the dataset -

  In this phase, we aim to collect a dataset that would consist of Covid-19 tweets. For pre-processing we would neeed to filter out the tweets which are specific to Covid-19 only.

- Phase 2: Dividing the dataset -

  In this second phase, in order to differentiate between the properties of fake news from that of real news, we will need to study the fake news and real news separately. Thus, we will divide the dataset into fake and real news by deploying a fact checking website.

- Phase 3: Creation of Graphs -

  Finally, once we have a dataset of fake and real news we would develop an algorithm that would be able to generate graphs for the fake and real news tweets involving the users and the data they share among themselves.

- Phase 4: Methodology involving supervised learning -

  In this stop we deploy a graph classification technique which would best suite for this type of problem by employing a classifier.

- Phase 5: Comparison with existing approaches -

  In order to compare our approach with the existing approach we would show the classification accuracy achieved in phase 4.

3.3 Dataset

We collect our dataset of fake and real news and the tweets associated with that news. We perform a filter on the content of the news to verify that it is related to Covid-19. Then we gather the re-tweeters and followers of this processed dataset in order to create a collection of graphs.

We collected the tweet Ids from the FakeNewsNet dataset [32, 33, 34] in March

2021 and then re-hydrated them to extract information regarding to their user Ids, re-tweeter Ids as well as the follower Ids of those tweeters. The real-world twitter dataset we collected after performing the data processing step has nearly 1000 nodes in fake news graph and 1000 nodes in real news graph. The properties of the fake news graphs are given in table 1.

Table 1: Overview of Fake News Graphs

Network Properties	**Values**
nodes	863
edges	866
directed?	False
weighted?	False
isolated nodes	1
self-loops	1
density	0.002326
min degree	0
max degree	29
avg degree	2.005794
degree assortativity	-0.449338
number of connected components	9
size of largest component	274 (31.75)%

Similarly, we collected the overview for the real news graph data and is given by table 2.

3.3.1 FakeNewsNet

The FakeNewsNet repository contains data that is extracted from PolitiFact and GossipCop fact checking websites. Further, each of these fact checking website have two set of news i.e., fake news and real news. Each of the fake and real news contains a json file for news-content which contains URL, text, images, keywords and several

Table 2: Overview of Real News Graphs

Network Properties	**Values**
nodes	1370
edges	1685
directed?	False
weighted?	False
isolated nodes	1
self-loops	0
density	0.001797
min degree	0
max degree	105
avg degree	2.459854
degree assortativity	0.393437
number of connected components	6
size of largest component	1283 (93.65)%

other factors associated with that news. From the text we can detect whether the news belongs to Covid-19 or not. Also, there is a list of tweets which have been made in response to content of each these news articles. From the tweet we extracted the tweet ID and user ID. Using the tweepy API and Python in Google Colab Notebook we were able to find the re-tweeters and followers of those tweet and re-tweets. This gives us a graph which could be used to detect the influencers using social network analysis. The architecture of the FakeNewsNet dataset is presented in figure 7.

Once the tweets are dehydrated from the tweets folder they have information related to the user ids, number of retweets, URls associated with it, images present in it if any, number of people who replied to the tweet and so on. The structure in which all this information is present is shown by the figure 8.

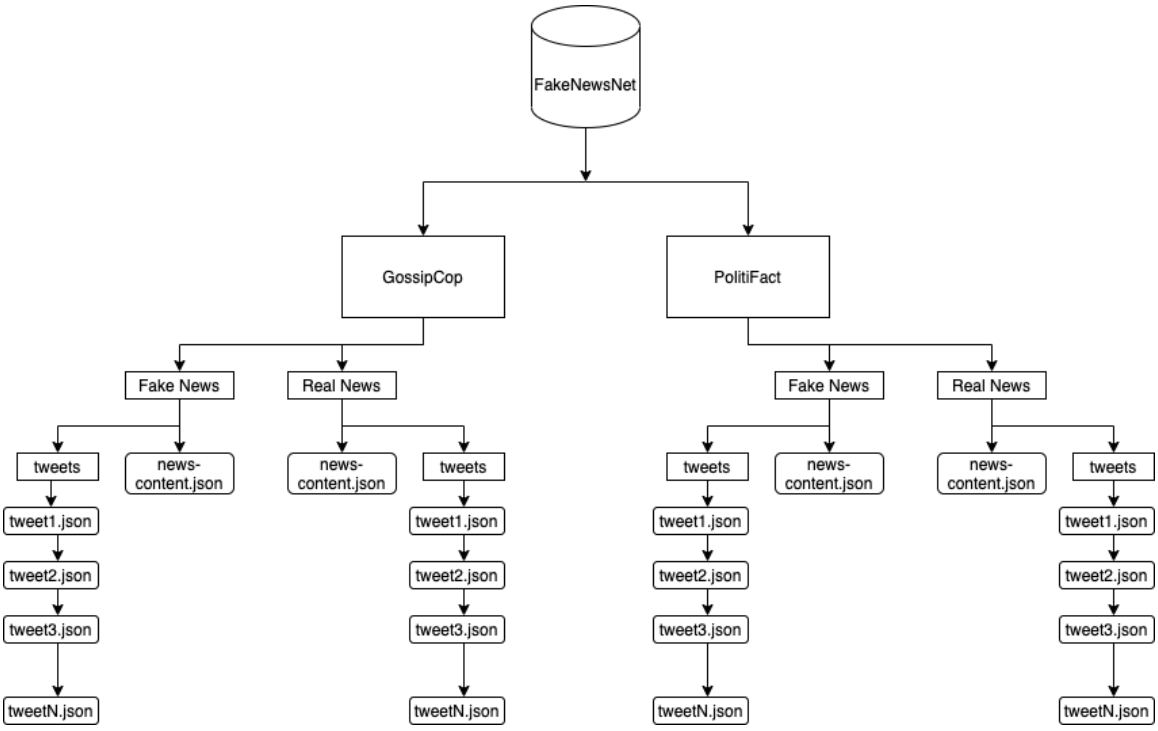

Figure 7: FakeNewsNet Architecture.

3.3.2 Data Pre-processing

After collecting the data from FakeNewsNet repository we performed a data pre-processing step. In this step we we ran our code through all the news-content.json files present in the Gossipcop and Politifact folders and collected only the news relevant to Covid-19. With this we filtered out the Twitter data that is specific for this research only. We further created a bag of words from all the news content collected and present them in the form of a word cloud as shown in figure 9.

As we can see in the word cloud of our collected dataset, the most frequently words mentioned are vaccine, covid-19 vaccine, year, news, pfizer, coronavirus vaccine, immune response, excess death. This shows the amount of anxiety, fear and stress that persists in the population for Covid-19 during this pandemic.

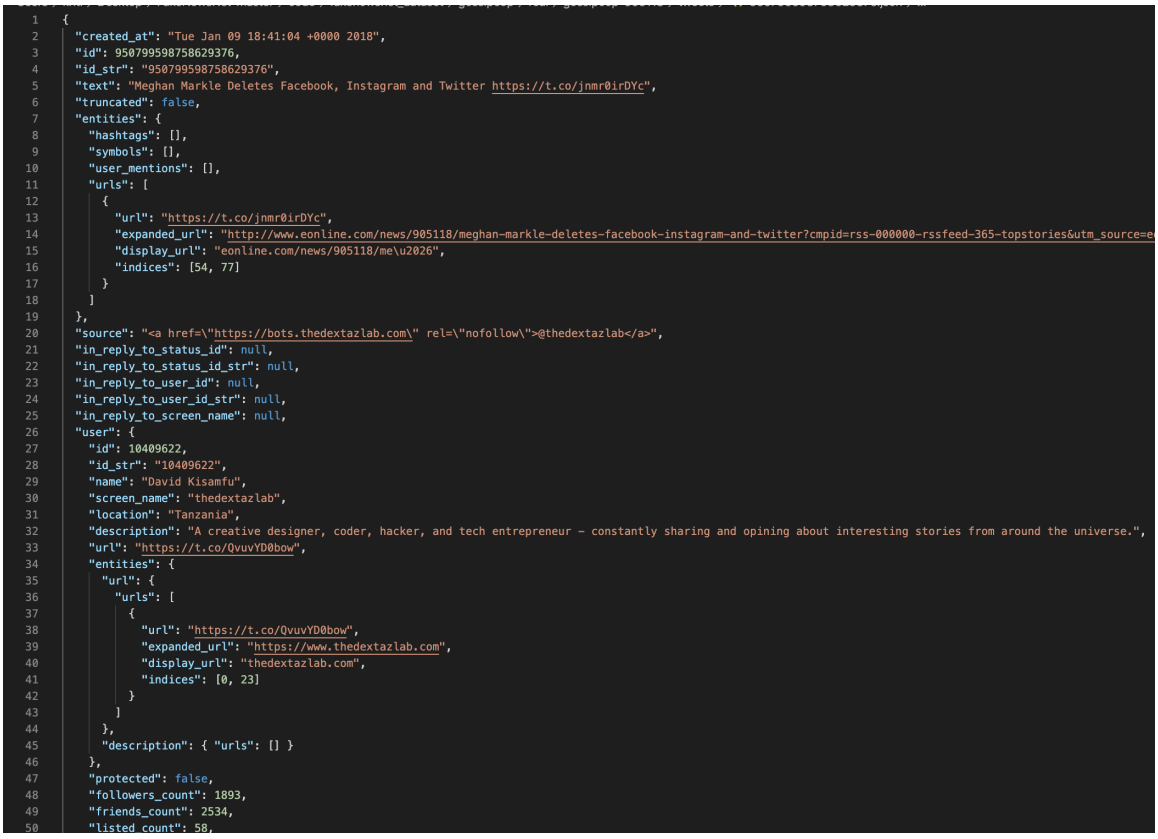

Figure 8: Tweet Description

3.4 Creation of the Graphs

After analyzing the tweet structure, we found that a tweet consists of several meaningful information such as the tweet ID, user ID of the user who tweeted the tweet as well as other factors such as URL of the images related to it, the text content, retweets associated with it. We created the graph using Breadth First Search (BFS) traversal. We visited to the tweets of each news content and then extracted the users who retweeted that tweet, thus, creating an edge between the user who tweeted and the user who retweeted. Further we fetched the followers of the re-tweeter to take into consideration all the people to whom this tweet for shared with. Figure 7 depicts the graph that was formed from the tweets, re-tweets and the followers.

The dataset we created consists of three comma separated text files,

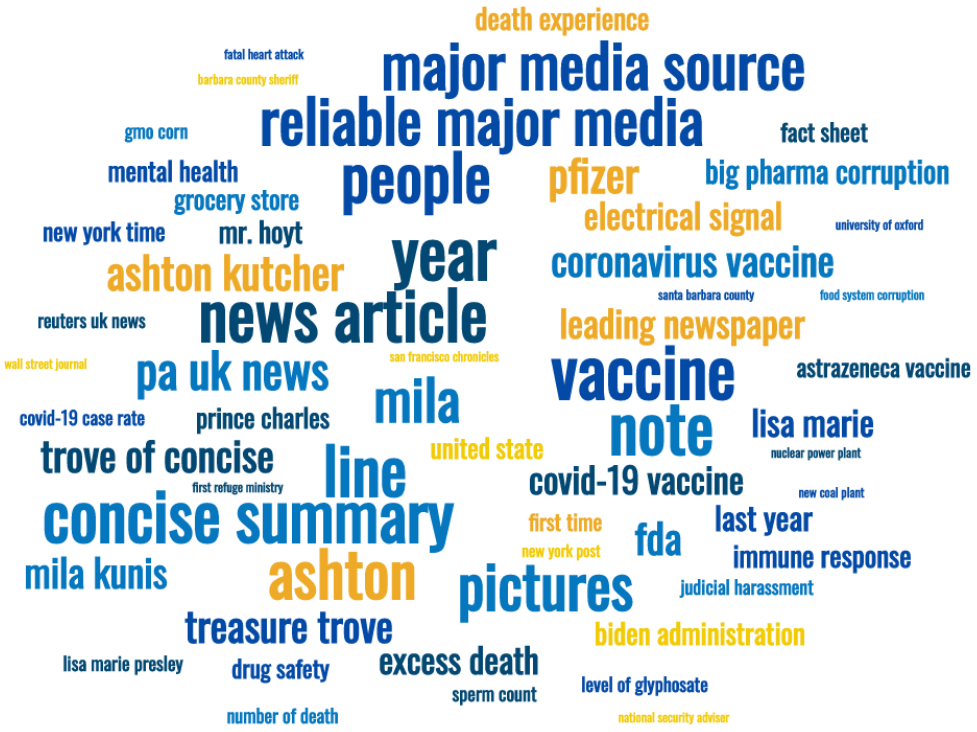

Figure 9: Word Cloud

COVID_A.txt, COVID_graph_indicator.txt and COVID_graph_labels.txt. This structure is followed from the benchmark dataset of [35].

- **COVID_A.txt:** The COVID_A.txt file is the adjacency matrix for all the graphs. It has edges separated by commas in the form (*node_id*, *node_id*).
- **COVID_graph_indicator.txt:** File COVID_graph_indicator.txt is a column vector which indicates the graph number for the node in the *ith* column.
- **COVID_graph_labels.txt:** The file, COVID_graph_labels.txt points to the label of the graph i in the *ith* column.

From algorithm 1 we can construct a twitter graph with nodes representing the user Ids and edges representing the connections between them. This algorithm is illustrated in the following steps.

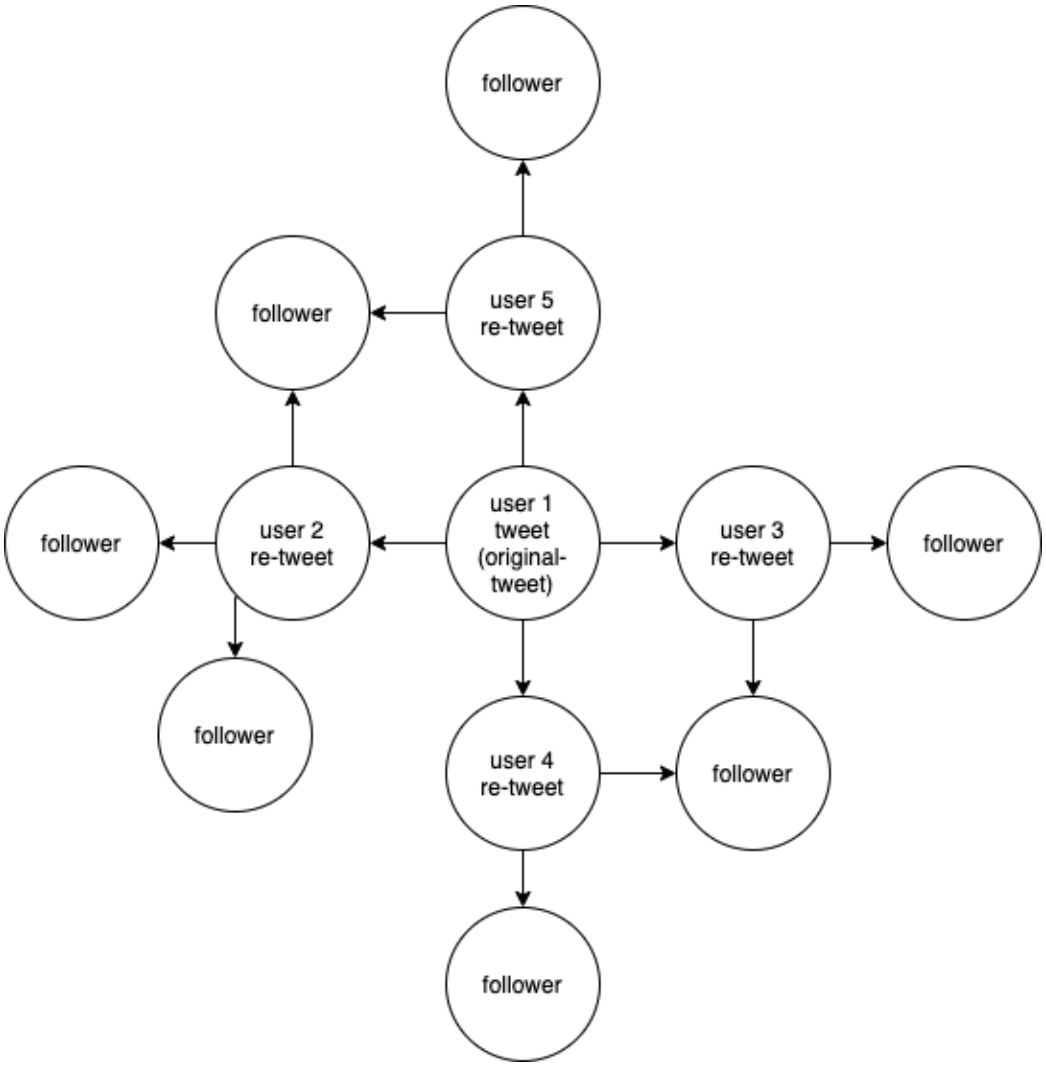

Figure 10: Tweet Graph.

Step 1: A Twitter user with $user\ Id:\ 1$ tweets a message with $tweet\ Id:\ 123$ among his/her Twitter network. We cash the $user\ Id:\ 1$ and $tweet\ Id:\ 123$ for collection of useful information from the data.

Step 2: With the $tweet\ Id:\ 123$ we find the list of $re-tweet\ Ids$ that have been made for this tweet.

Step 3: Then we start iterating through this list of $re-tweet\ Ids$ and start retrieving

Algorithm 1 Twitter Graph Creation

```
1: procedure CreateGraph(rootTweetId, rootUserId)
2:     Initialize visitedTweets as empty array
3:     Initialize graphEdges
4:     Append rootTweetId, rootUserId in Queue
5:     while length of Queue ≠ 0 do
6:         collect reTweetersIds for rootTweetId
7:         for reTweeterId in reTweetersIds do
8:             add rootUserId, reTweeterId to graphEdges
9:         if reTweetersIds = None then
10:            collect followers of rootUserId
11:            for follower in followers do
12:                add rootUserId, follower to graphEdges
13:        Append reTweetersId to Queue
14:        visitedTweets ← rootTweetId
```

there respective $re-tweeter\ Ids$ who re-tweeted this tweet.

Step 4: Once we have the list of $userIds$ who tweeted the tweet and $re-tweeter\ Ids$ who re-tweeted the tweet we can store an edge between them considering as source and target respectively.

Step 5: After all the $re-tweeter\ Ids$ are collected from the tweet, we collect the list of $follower\ Ids$ of the followers who are following those re-tweeters. This gives us a better picture of all the users involved in viewing the original tweet with $tweet\ Id:\ 123$ made by the user with $user\ Id:\ 1$.

3.4.1 Time Complexity Analysis

Considering E number of edges and V number of vertices or nodes in a single Tweet graph, the time complexity of algorithm 1 would be $O(V + E)$. This is similar to the time complexity of breadth first search algorithm in the worst case scenario since every edge and every node is being visited in the graph. In a real time situation there are other factors as well that affects the computation of this algorithm. One such factor is the Twitter rate limit of re-hydrating tweets and collecting information

such as re-tweeters and followers from a tweet Id. For a standard developer account Twitter allows only 15 requests per rate limit window, and then it allows 15 requests per window per access token. Due to this reason it can take up to two hours to develop only 20 graphs of 20 different tweets even after applying caching. However, this time can be skipped by gaining access to the Twitter premium account.

3.5 Weisfeiler-Lehman Kernel Approach

One of the most important techniques used for solving the graph classification problem is by using graph kernels. Graph kernels reduce the dimensionality by combining the neighbors of a node and renaming the labels of set of these nodes at each step.

3.5.1 Computation

The WL kernel approach computes the isomorphism test between two graphs. The way it reduces the dimentionality at each step is explained by the figure 11. For computing the WL kernel method we process simultaneously all the given N graphs and then perform the following steps.

Algorithm 2 Weisfeiler-Lehman Kernel Algorithm

Step 1: For all the N graphs compute the multiset label l_i.
Step 2: Collect the immediate neighboring nodes and then concatenate their labels into a single string.
Step 3: Perform the label compression by mapping the neighboring strings into one label.
Step 4: In the final step, relabeling is done for all the nodes in the graph.

Given two labeled graphs G and G', in the 1st iteration step 1 will determining the multiset- label and step 2 will perform sorting. Further step 3 would would perform the label compression followed by relabeling in step 4. Figure 11 visualizes the steps occurring at each iteration of the Weisfeiler Lehman graph kernel method.

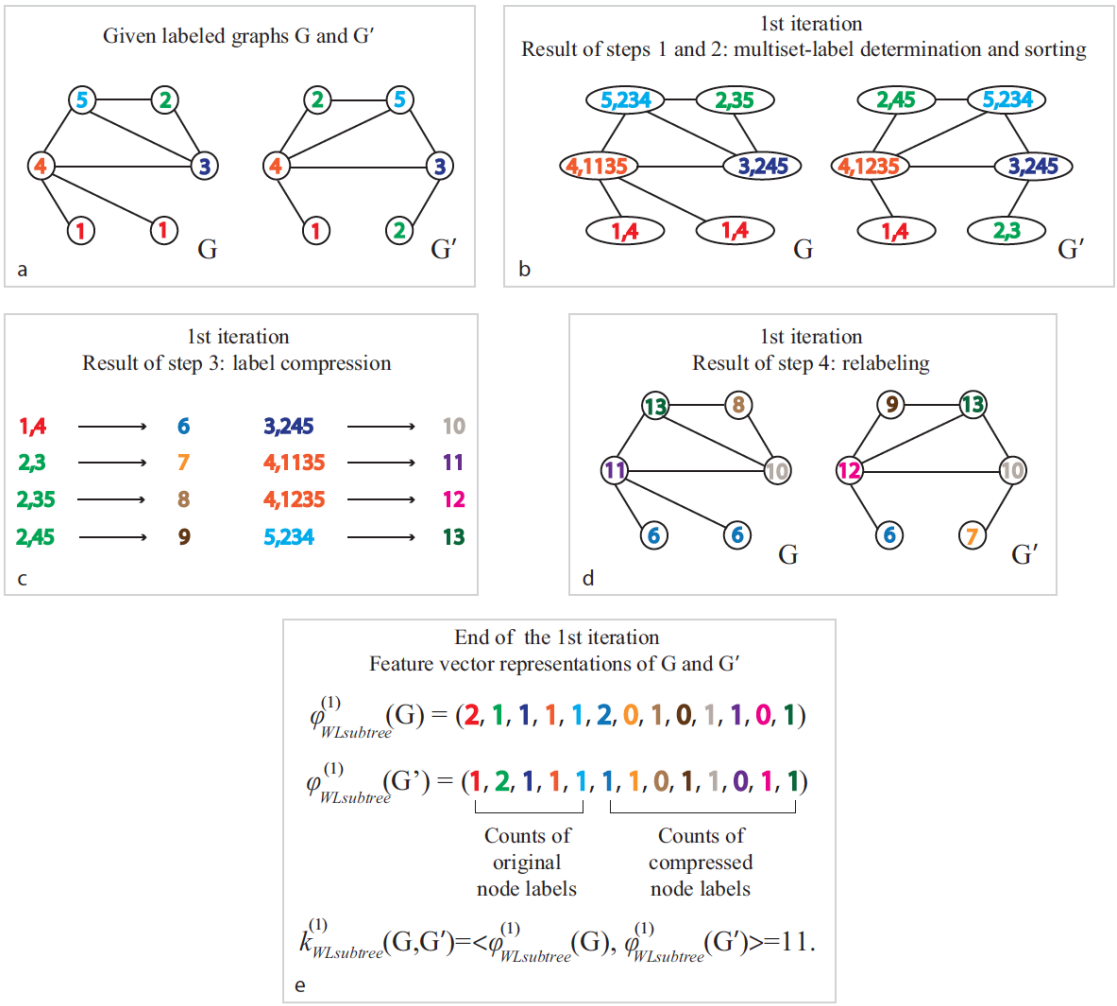

Figure 11: Steps Involved in the Computation of Weisfeiler-Lehman Kernel [3]

3.5.2 Time Complexity Analysis

The Weisfeiler-Lehman (WL) graph kernel is a supervised learning approach for the graph classification problem. In order to perform classification, a matrix is sent to a kernel-based machine learning algorithm as input. Further, an isomorphism WL test is performed between two graphs, so as to categorize them into different categories. If n represents the number of iterations then $O(n)$ operations are required for labeling. Therefore, the time complexity analysis for this WL isomorphism test is $O(nt)$, where,

n represents the iterations and t represents the time taken to compress each label in the iterations.

CHAPTER 4

Experimental Evaluation

In order to distinguish between the fake news graph from the real news graph, we look into some properties. The definitions of these properties are as follows:

1. Density -

   This parameter measures the number of connections that exists between nodes in comparison to the number of connections that could be possible between nodes.

2. Min degree -

   In a graph, the min degree is the measure of the number of nodes with the least number of edges connected to a node.

3. Max degree -

   In a graph, the max degree is the measure of the number of nodes with the greatest number of edges connected to a node.

4. Avg degree -

   In a graph, the avg degree is the measure of the number of the nodes as compared to the number of edges present in it.

5. Degree assortativity -

   This metric measures the the extent to which nodes associate among themselves in a graph. Therefore, it depicts how much nodes of high degree are associated with other nodes of high degree and respectively for the lower nodes as well.

6. Degree distribution -

   In a network of nodes, the degree distribution is a metric which determines the

probability distribution of the degrees of these nodes over the whole network of nodes.

7. Page ranking -

   This is an algorithm which can measure the importance of each of the nodes within a given graph.

8. Size of the largest component -

   This is a critical metric in terms of analyzing the propagation of news in a network. This measures the extent to which a news has been spread. It can help to distinguish whether real news or fake news has more tendency to spread to a larger group of people.

From table 1 and table 2, we analyze that the density for fake news is slightly higher than the real news graph. Also, we see that the size of the largest component for real news (approximately 90%) is three times larger than the largest component for fake news (approximately 30%). This means that fake news tends to propagate among a small group of people, whereas, true news tend to propagate among a larger community. The degree assortativity for fake news is negative, while the degree assortativity for real news is positive. This further, proves our point that in real news propagation the nodes with high degrees (respectively low degrees) tends to connect with the nodes of high degrees (respectively low degrees) whereas, in fake news the nodes with high degrees (respectively low degrees) tends to connect with nodes of low degrees (respectively high degrees). Above all a fake news differs from real news in the sense that it keeps on circulating among a small community while real news vastly reaches out to a larger community.

Figure 12 gives the statistics for community detection for fake news graph community detection and the figure 13 gives the statistics for real news graph community

detection. From the graphs we analyze that the communities present in real news is almost double the number of communities present in fake news, even though both the graphs have almost same number of nodes.

```
------------------------------------------------
Fake news community detection:
------------------------------------------------
PLM(balanced,pc,turbo) detected communities in 0.0005021095275878906 [s]
solution properties:
-------------------  ---------
# communities        39
min community size    1
max community size   56
avg. community size  22.1282
modularity            0.931029
-------------------  ---------
```

Figure 12: Fake Community Detection Analysis

```
------------------------------------------------
Real news community detection:
------------------------------------------------
PLM(balanced,pc,turbo) detected communities in 0.0005574226379394531 [s]
solution properties:
-------------------  ---------
# communities        60
min community size    1
max community size   49
avg. community size  22.8333
modularity            0.931744
-------------------  ---------
```

Figure 13: Real Community Detection Analysis

Next, we look into the degree distribution for each network. Figure 14 shows the degree distribution for fake news graph. And on a similar note, figure 15 shows the degree distribution for real news graph. We observe a steep decline in the number of nodes in the case of fake news network. Whereas, we observe a gradual decline in the number of nodes in the case of real news network. This means that in fake news has either an abundance of extreme in-degree node or an abundance of extreme out-degree node. However, this is not the case with real news graphs. In real news the nodes are

spread uniformly between the nodes having the maximum in-degree and the nodes having the maximum out-degree.

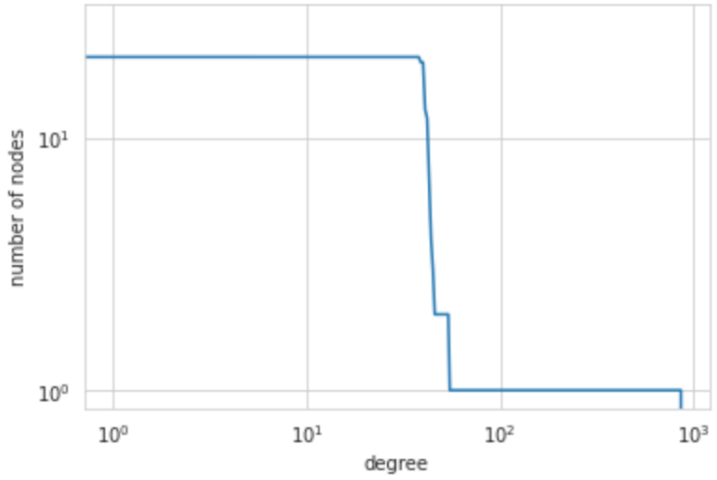

Figure 14: Degree Distribution for Fake News

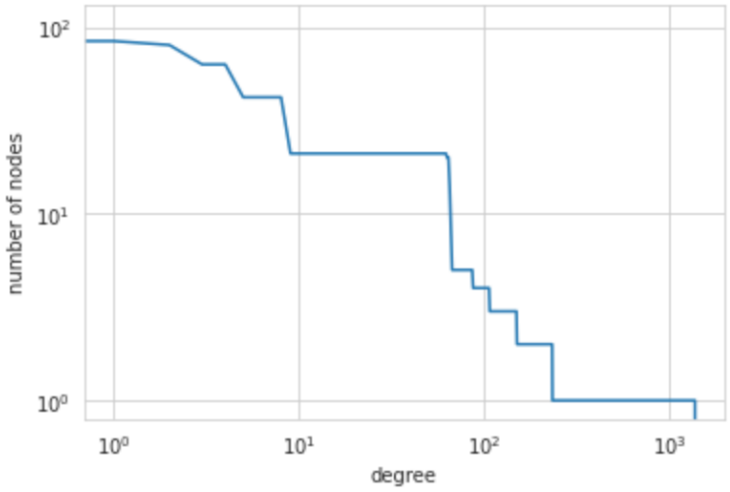

Figure 15: Degree Distribution for Real News

Further we observe the page ranking statistics for fake news graph of the ten most central nodes in figure 16. Similarly, in figure 17 we observe the page ranking statistics for real news graph of the ten most central nodes. The difference between

the values of page ranking of fake and real news is that each node in fake news graph is ranked ten times higher than each node of real news graph.

```
[(483, 0.0011587717278548 87),
 (440, 0.0011587711484811 9),
 (441, 0.0011587711484811 9),
 (800, 0.0011587709553566243),
 (801, 0.0011587709553566243),
 (802, 0.0011587709553566243),
 (23, 0.001158770665669785),
 (24, 0.001158770665669785),
 (25, 0.001158770665669785),
 (26, 0.001158770665669785)]
```

Figure 16: Page Ranking for Fake News

```
[(857, 0.00072994160635761 99),
 (1349, 0.00072994160635761 99),
 (6, 0.0007299409220519227),
 (9, 0.0007299409129277995),
 (7, 0.0007299409038036765),
 (15, 0.0007299409038036765),
 (605, 0.0007299408946795534),
 (609, 0.0007299408946795534),
 (2, 0.0007299408855554304),
 (4, 0.0007299408855554304)]
```

Figure 17: Page Ranking for Real News

Figure 18 shows the visualization of the propagation of fake news through Twitter tweets among the individuals and the society. Also, figure 19 shows a visualization of the propagation of real news through Twitter tweets among the individuals and the society. From the graph visualizations we observe that fake news graphs are more

concentrated, indicating small and local communities. However, the real news graphs have larger communities and the news spread more far and wide in these communities.

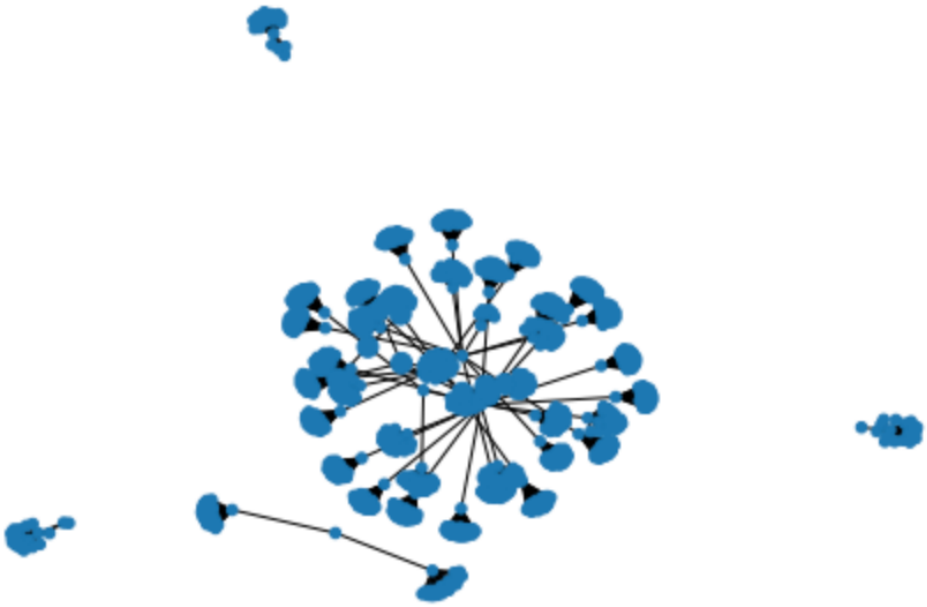

Figure 18: Graph Visualization for Fake News Propagation

4.1 Graph classification results

As a baseline for our research we make use of the graph kernel approach for graph classification. Although there exists several other graph kernel approaches such as, Shortest-path, random walk, weighted decomposition, optimal assignment, however, we choose to perform the Weisfeiler-Lehman graph kernel approach because of it has the property of performing isomorphism test in between any two given graphs. This isomorphism test will distinguish the fake news graph from the real news graph and help in calculating a prediction accuracy for our research.

4.1.1 Classifiers

We can perform classification with classifiers such as Support Vector Machine (SVM), K-Nearest Neighbors (K-NN), AdaBoost, random forest, Multilayer Perceptron

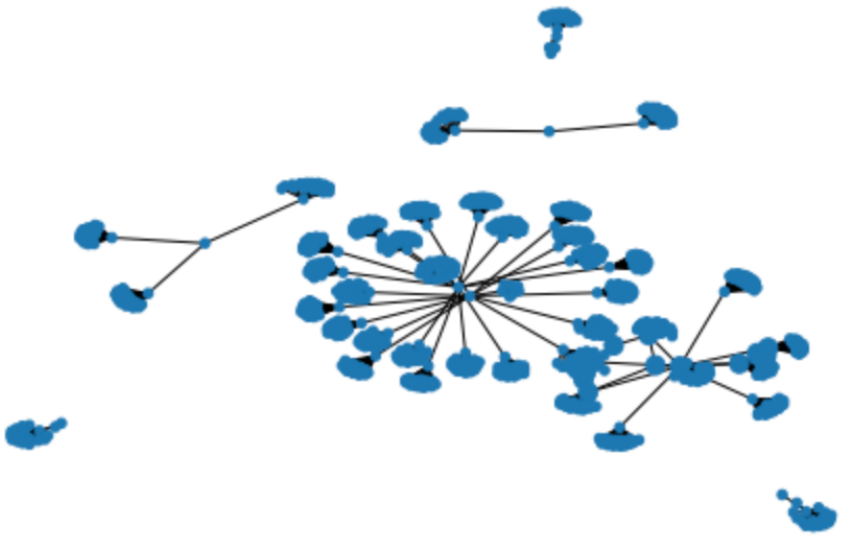

Figure 19: Graph Visualization for Real News Propagation

(MLP), Gaussian Naive Bayes, Decision Tree (DTree) and Logistic Regression (LR). These classifiers are defines as follows:

1. Support Vector Machine (SVM) Classifier -

   It is a supervised learning model, and it performs very efficiently on both linear and non-linear classification by employing the kernel trick. It separates the input graphs into different classes by dividing them with a hyperplane as shown in figure 20.

2. K-Nearest Neighbor (KNN) Classifier -

   This classifier classifies an object based on the its frequency among given k samples during training. The category assigned to an object is based on the category of the majority of objects it is surrounded with. In figure 21 the green object would be classified as red if k = 3 because among the 3 closest objects red is in majority. If the value of k would be 5 then blue is in majority and the

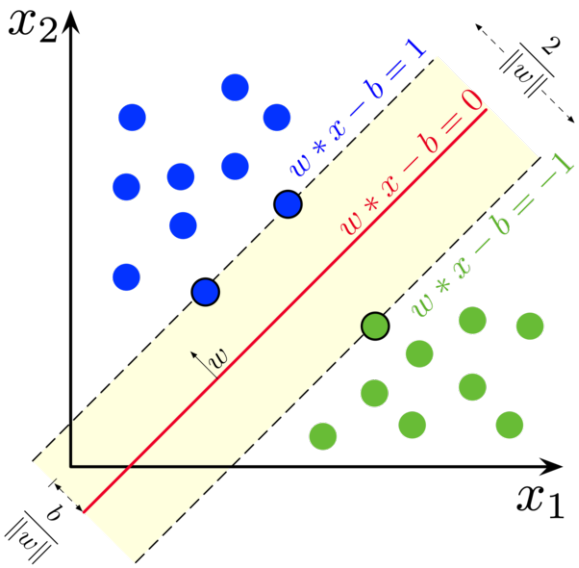

Figure 20: SVM Trained From Samples of Two Classes [4]

green object would be classified as blue.

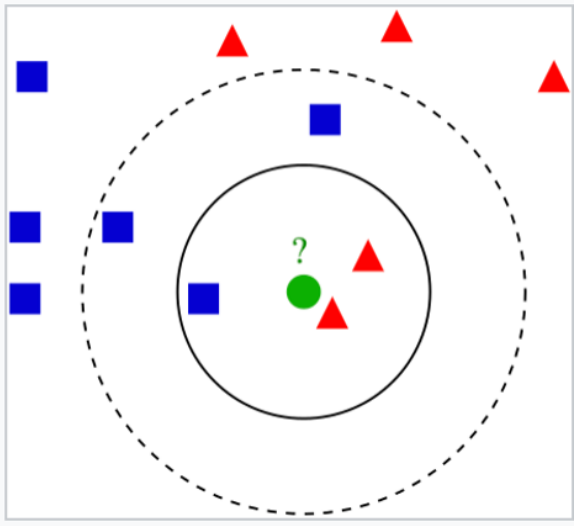

Figure 21: Example of KNN Classification [5]

3. Decision Tree (DTree) Classifier -

   A decision tree has a flowchart-like structure. It comprises of three types of nodes, the decision node, the chance node and the end node. A decision tree learns certain decision rules to form an outcome which is then represented by

the leaf nodes. The classification rules in the decision tree are represented by the paths from the root to the leaf of the decision tree.

4. AdaBoost Classifier -

   This classifier tries to boost the performance of several weak classifiers such as Decision Tree classifier by combining them. By the boosting a weak classifier it leverages the performance of the model.

5. Random Forest Classifier -

   This classifier is trained by combining several decision trees together as shown in figure 22. In this algorithm the performance of the model enhances by bagging together many decision trees. The majority of the votes of the decision trees defines the final category of the graph.

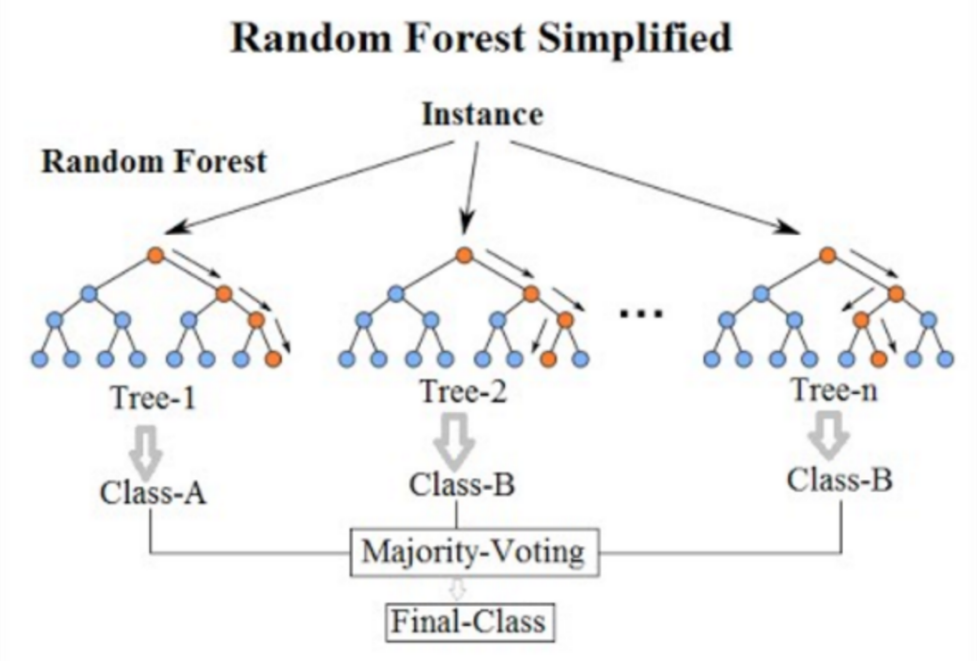

Figure 22: Random Forest Classification [6]

6. Multilayer Perceptron (MLP) Classifier -

It is a feedforward neural network classifier. It has a linear activation function which maps the input to each neuron's output. It has an input layer, one or more hidden layers and an output layer.

7. Gaussian Naive Bayes Classifier -

   This model works best for continuous values and when features have a normal distribution. It preordains that each feature is independent of each other. This model can be trained very easily as it can perform classification by training on a small amount of data.

8. Logistic Regression (LR) Classifier -

   This classifier is based on the logistic sigmoid function to transform the output into a probability value. This probability value can further be mapped into different classes.

Next, we define the steps involved in computing the WL subtree kernel with the help of using each of these classifiers.

4.1.2 Steps for performing WL subtree kernel approach

The steps involved in performing machine learning on the given collection of graphs with WL kernel approach are as follows:

1. Initializing the WL subtree kernel framework -

   This is a framework given by the Grakel library to implement the WL kernel more efficiently and effectively. In this step we initialize the WL framework in addition to the vertex histogram kernel.

2. Fetching dataset -

   In this step, we retrieve the data from the text files of nodes, edges and graphs that we created earlier in section 3.4. We create two matrices G which is a 2

dimensional matrix and y which is a 1 dimensional matrix using the numpy library in python. G contains the data for edges and graph indicators, while, y contains the data for graph classification and labels.

3. Splitting the dataset into training and testing -

   Splitting the data into training and testing is the most crucial step of classification. We utilize the scikit-learn library in python to split the data into training and testing with the train_test_split function.

4. Initializing cross validation -

   In this step, we initialize a k-fold cross validation for the model on 20% of the test dataset. It takes k observations from the complete data to test on it iteratively.

5. Employ a Classifier -

   Here we perform the classification by employing each of the classifiers mentioned in section 4.1.1. First we train the classifier on the given training dataset. Secondly, predictions are made on the test dataset.

6. Calculating classification accuracy -

   Finally, in this step we compute a mean classification accuracy for the different k values ranging from 2 to 10.

After implementing the above steps, we obtain an a mean classification accuracy and table 3 gives a brief idea of how these classifiers performed on the given data set by showing their mean classification accuracies. The classifiers random forest, linear regression, MLP and Gaussian Naive Bayes were over-fitting for the given dataset so a mean classification accuracy could not be achieved for these classifiers. However, the

classifiers Ada Boost, K-NN, decision tree and SVM gave a competitive classification accuracy for k-fold cross validation and values of 2, 3, 4 and 9.

Table 3: Graph Classification Accuracy

Classifier	**k = 2**	**k = 3**	**k = 4**	**k = 9**
Ada Boost	over-fitting	over-fitting	**93.75**	over-fitting
K-NN	60.72	61.67	45.83	**77.79**
Decision Tree	91.67	over-fitting	**93.75**	over-fitting
SVM	**46.43**	45	37.5	38.89

From the graph classification accuracies we observe that Ada Boost was mostly over-fitting, however, for 4-fold cross validation it gave 93.75% as mean classification accuracy. Decision tree on the other hand performed very similarly to Ada Boost, but it was less over-fitting as compared to Ada Boost. It also achieved the highest mean classification accuracy of 93.75% during 4 fold cross validation. K-NN and SVM outperformed Ada Boost and Decision tree in terms of over-fitting and between K-NN and SVM, we observe that K-NN outperforms SVM. The highest mean classification accuracy that SVM achieved is 46.43%, whereas, the highest mean classification accuracy that K-NN achieved is 77.79%. So, after analyzing the results, we concur that K-NN outperforms all the other classifiers by gaining the mean classification accuracy of nearly 78%.

CHAPTER 5

Conclusions and Future Work

To conclude, in this research we collected and created a new dataset. Starting from the FakeNews repository [33, 34, 32], from Twitter, we filter the COVID-19 related topics, and modeled them with graphs. In our approach we try to answer the important question of how real news propagate over how fake news propagate and what are their differences and similarities. Fake news pose a problem since they can alter the public's opinion during crisis such as the Covid-19 pandemic. In order to get more insight on how fake news is spread compared to real news we perform social network analysis and reformulate the problem as a graph classification problem. The created dataset is modelled as follows: news (fake or real) are graphs, users are nodes, and edges are the flow of information between retweeters or followers. Some of the analysis includes characteristics of the graphs of each category (fake and real), and applying community detection to detect similarities and differences between the real news and fake news, and finding the influencers (user of original tweet) in each community. Further we performed graph classification on these graphs, that have a label fake or real, with the Weisfeiler Lehman kernel approach to get an average classification accuracy of 93.75% over the different cross fold validations.

In the future, more work can be done on collecting useful data from other social media platforms such as Facebook, Instagram and TikTok, and a comparison can be made how real versus fake news is being spread through these different social media sites. It will help to get better understanding of which which social media platform an individual or the society itself gets influenced by the spread of any kind of piece of information.

LIST OF REFERENCES

[1] J. Kluger, ‘‘Accidental poisonings increased after president trump’s disinfectant comments,’’ *Time*, May 2020. [Online]. Available: https://time.com/5835244/accidental-poisonings-trump/

[2] D. M. Blei, A. Y. Ng, and M. I. Jordan, ‘‘Latent dirichlet allocation,’’ *J. Mach. Learn. Res.*, vol. 3, no. null, p. 993–1022, Mar. 2003.

[3] N. Shervashidze, P. Schweitzer, E. J. van Leeuwen, K. Mehlhorn, and K. M. Borgwardt, ‘‘Weisfeiler-lehman graph kernels,’’ *Journal of Machine Learning Research*, vol. 12, no. 77, pp. 2539–2561, 2011. [Online]. Available: http://jmlr.org/papers/v12/shervashidze11a.html

[4] W. contributors, ‘‘Support-vector machine,’’ May 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Support-vector_machine&oldid=1021890669

[5] W. contributors, ‘‘K-nearest neighbors algorithm,’’ May 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=K-nearest_neighbors_algorithm&oldid=1021698330

[6] W. contributors, ‘‘Random forest,’’ May 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Random_forest&oldid=1021839899

[7] G. Pennycook, J. McPhetres, Y. Zhang, J. G. Lu, and D. G. Rand, ‘‘Fighting covid-19 misinformation on social media: Experimental evidence for a scalable accuracy-nudge intervention,’’ *Psychological science*, vol. 31, no. 7, p. 770–780, 2020.

[8] A. Press, ‘‘Facebook to label misleading politicians’ posts, including trump’s,’’ *The Los Angeles times*, Jun 2020. [Online]. Available: https://www.latimes.com/business/technology/story/2020-06-26/facebook-following-twitter-will-label-posts-that-violate-its-rules-including-trumps

[9] W. Ahmed, J. Vidal-Alaball, J. Downing, and F. López Seguí, ‘‘Covid-19 and the 5g conspiracy theory: Social network analysis of twitter data,’’ *Journal of medical internet research*, vol. 22, no. 5, p. e19458, 2020.

[10] L. Singh, S. Bansal, L. Bode, C. Budak, G. Chi, K. Kawintiranon, C. Padden, R. Vanarsdall, E. Vraga, and Y. Wang, ‘‘A first look at covid-19 information and misinformation sharing on twitter,’’ *ArXiv*, 2020.

[11] N. Rosenfeld, A. Szanto, and D. C. Parkes, ''A kernel of truth: Determining rumor veracity on twitter by diffusion pattern alone,'' in *Proceedings of The Web Conference 2020.* ACM, 2020.

[12] B. Rath, A. Salecha, and J. Srivastava, ''Detecting fake news spreaders in social networks using inductive representation learning,'' 2020. [Online]. Available: http://arxiv.org/abs/2011.10817

[13] S. Vosoughi, D. Roy, and S. Aral, ''The spread of true and false news online,'' *Science (New York, N.Y.)*, vol. 359, no. 6380, p. 1146–1151, 2018.

[14] [Online]. Available: https://coronavirus.jhu.edu/map.html

[15] D. Garcia-Gasulla, S. A. Napagao, I. Li, H. Maruyama, H. Kanezashi, R. P'erez-Arnal, K. Miyoshi, E. Ishii, K. Suzuki, S. Shiba, and et al., ''Global data science project for covid-19 summary report,'' 2020. [Online]. Available: http://arxiv.org/abs/2006.05573

[16] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, ''Bert: Pre-training of deep bidirectional transformers for language understanding,'' 2018. [Online]. Available: http://arxiv.org/abs/1810.04805

[17] P. Qi, Y. Zhang, Y. Zhang, J. Bolton, and C. D. Manning, ''Stanza: A python natural language processing toolkit for many human languages,'' in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations.* Association for Computational Linguistics, 2020.

[18] T. Matsumoto, W. Sunayama, Y. Hatanaka, and K. Ogohara, ''Data analysis support by combining data mining and text mining,'' in *2017 6th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI)*, 2017, pp. 313–318.

[19] G. B. Mangmang, L. Feliscuzo, and E. A. Maravillas, ''Descriptive feedback on interns' performance using a text mining approach,'' in *2019 14th International Joint Symposium on Artificial Intelligence and Natural Language Processing (iSAI-NLP)*, 2019, pp. 1–4.

[20] P. Jayasekara and A. K.S., ''Text mining of highly cited publications in data mining,'' in *2018 5th International Symposium on Emerging Trends and Technologies in Libraries and Information Services (ETTLIS)*, 2018, pp. 128–130.

[21] E. Reategui, M. Klemann, and M. D. Finco, ''Using a text mining tool to support text summarization,'' in *2012 IEEE 12th International Conference on Advanced Learning Technologies*, 2012, pp. 607–609.

[22] Q. Wu, X. Deng, C. Zhang, and C. Jiang, ''Lda-based model for topic evolution mining on text,'' in *2011 6th International Conference on Computer Science Education (ICCSE)*, 2011, pp. 946–949.

[23] A. Mahmood and M. Small, ‘‘Subspace based network community detection using sparse linear coding,’’ *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 3, pp. 801–812, 2016.

[24] S. R. Chintalapudi and M. H. M. K. Prasad, ‘‘A survey on community detection algorithms in large scale real world networks,’’ in *2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom)*, 2015, pp. 1323–1327.

[25] M. Ross, C. A. Graves, J. W. Campbell, and J. H. Kim, ‘‘Using support vector machines to classify student attentiveness for the development of personalized learning systems,’’ in *2013 12th International Conference on Machine Learning and Applications*, vol. 1, 2013, pp. 325–328.

[26] K. Pahwa and N. Agarwal, ‘‘Stock market analysis using supervised machine learning,’’ in *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, 2019, pp. 197–200.

[27] C.-J. Huang, M.-C. Liu, S.-S. Chu, and C.-L. Cheng, ‘‘Application of machine learning techniques to web-based intelligent learning diagnosis system,’’ in *Fourth International Conference on Hybrid Intelligent Systems (HIS'04)*, 2004, pp. 242–247.

[28] A. Mahboubi, L. Brun, and F.-X. Dupé, ‘‘Object classification based on graph kernels,’’ in *2010 International Conference on High Performance Computing Simulation*, 2010, pp. 385–389.

[29] M. Borhani and H. Ghassemian, ‘‘Hyperspectral image classification based on spatial graph kernel,’’ in *2014 22nd Iranian Conference on Electrical Engineering (ICEE)*, 2014, pp. 1811–1816.

[30] M. G. Seenappa, K. Potika, and P. Potikas, ‘‘Short paper: Graph classification with kernels, embeddings and convolutional neural networks,’’ in *2019 First International Conference on Graph Computing (GC)*, 2019, pp. 88–93.

[31] M. Neumann, R. Garnett, C. Bauckhage, and K. Kersting, ‘‘Propagation kernels: efficient graph kernels from propagated information,’’ *Machine learning*, vol. 102, no. 2, p. 209–245, 2016.

[32] K. Shu, D. Mahudeswaran, S. Wang, D. Lee, and H. Liu, ‘‘Fakenewsnet: A data repository with news content, social context and dynamic information for studying fake news on social media,’’ *arXiv preprint arXiv:1809.01286*, 2018.

[33] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, ‘‘Fake news detection on social media: A data mining perspective,’’ *ACM SIGKDD Explorations Newsletter*, vol. 19, no. 1, pp. 2236, 2017.

[34] K. Shu, S. Wang, and H. Liu, ‘‘Exploiting tri-relationship for fake news detection,’’ *arXiv preprint arXiv:1712.07709*, 2017.

[35] K. Kersting, N. M. Kriege, C. Morris, P. Mutzel, and M. Neumann, ‘‘Benchmark data sets for graph kernels,’’ 2016. [Online]. Available: http://graphkernels.cs.tu-dortmund.de